MURE: MCNP Utility for Reactor Evolution - Description of the methods, first applications and results

O. Méplan, A. Nuttin, O. Laulan, S. David, F. Michel-Sendis, J. Wilson

ENC 2005
Paper no: XXXX

MURE : MCNP UTILITY FOR REACTOR EVOLUTION
DESCRIPTION OF THE METHODS, FIRST APPLICATIONS AND RESULTS

Olivier MEPLAN, meplan@lpsc.in2p3.fr
Alexis NUTTIN and **Olivier LAULAN (LPSC Grenoble, France)**
Sylvain DAVID, Franco MICHEL-SENDIS and **Jonathan WILSON (IPN Orsay, France)**

Introduction

The main aim of the MURE package is to perform nuclear reactor time-evolution using the widely used particle transport code MCNP [1], a Monte Carlo code that is mostly written in FORTRAN. MURE is based on C++ objects allowing a great flexibility in the use. There are three main parts in this library.

- Part 1 : Definition of the geometry, materials, neutron source, tallies, ...
- Part 2 : Construction of the nuclear tree, the network of links between neighbour nuclei via radioactive decays and nuclear reactions.
- Part 3 : Evolution of some materials, by solving the corresponding Bateman's equations.

Moreover an interface[2] to NJOY[3] in order to process cross-sections at the wanted temperature is provided.

Part 1 can be used independently of the two others ; it allows “easy” generation of MCNP input files by providing a set of classes for describing complex geometries. The ability to make quick global changes to reactor component dimensions and the ability to create large lattices of similar components are two important features that can be implemented by the C++ interface. It should be noted that some knowledge of MCNP is very useful in understanding the geometry generation philosophy.

Part 2 builds the specific nuclear tree from an initial material composition (list of nuclei). The tree of each “evolving” nucleus is created by following the links between neighbours via radioactive decay and/or reactions until a self-consistent set of linked nuclei is extracted. Nuclei with half-lives which are very much shorter than the evolution time steps are removed from the tree and their parents and daughters re-linked in the correct way.

Part 3 aims at simulating the evolution of the fuel within a given reactor over a time period of up to several years, by successive steps of MCNP calculation and numerical integration of Bateman's equations. Each time MCNP is called, the reactor fuel composition will have changed due to the reaction/decay process occurring inside. Changes in geometry, temperature, external feeding or extraction during the evolution can also be taken into account. Obviously this part is not independent of the 2 others (Figure 1).

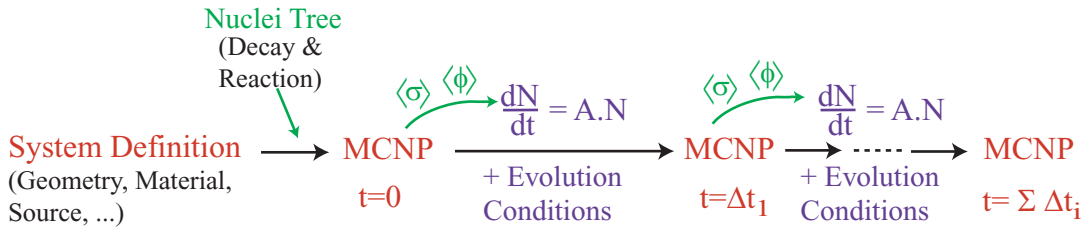

Figure 1: General scheme of an evolution calculation in MURE.

These main three parts of MURE are described in principles in the first section of this paper. The second one will then give first results obtained on precise simulation cases: FNR benchmark, PWR Th/Pu evolution and MSR temperature calculation.

1. Main MURE functions: geometry generation, nuclear data management and evolution calculation

Easy MCNP generation by means of a few intuitive C++ classes

The definition of the precise geometry of a system (reactor, experimental setup) for MCNP is generally a heavy task if the system is complex ; moreover, in such case, the modification of an existing geometry requires more or less a complete rewriting of the MCNP input file. The MURE interface provides tools to do

this in a very easy way using objects. Following the MCNP philosophy, main classes to define a system are *Shape*, *Cell* (a *Shape*+a *Material*), *Material*, *MCNPSource* and *Tally*.

Shapes are geometric shapes of a system part ; there are 4 basic *Shapes*, namely *Sphere*, *Brick*, *Cylinder* (infinite) and *Plane* (half space) . Then one can define *Nodes* which are intersections and unions of basic shapes. Two useful *Nodes* are already defined: *Tube* (finite pipe with a given thickness) and *Hexagon* (finite or infinite). When a *Shape* is defined, it could be translated, rotated or placed in another *Shape*. The following example shows how readable the geometry generation is:

```
Shape_ptr S(new Sphere(R,x0,y0,z0);
Shape_ptr B(new Brick(a/2,a/2,a/2));
S->Translate(dx,dy,dz);
B->Rotate(phi,theta,psi);
S>>B;
```

The last line means that the *Sphere* is put into the *Brick* (see Fig. 2 for generic example of the convenient ">>" operator).

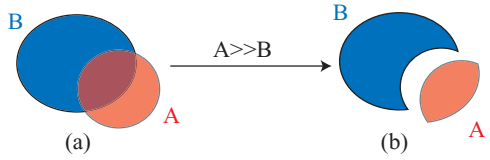

Figure 2: Placement of a *Shape* in another.

A *Cell* associates a *Shape* with a *Material* ; Lattice cells could be done for hexahedra lattice or hexagonal one. Materials are defined by giving the density and adding nuclei (identified by their Z and A) with a proportion:

```
Material *H2O=new Material(1.0);  // density in g/cm3
H2O->SetTemperature(600);         // temperature K
H2O->AddNucleus(1,1,2.);
H2O->AddNucleus(8,16,1.);

Cell *brick=new Cell(B,H2O);
```

To be noticed that the MCNP material extension can be extracted automatically. Tallies are taken into account via the *Tally* class but only the cell flux tallies are fully implemented, with, in particular, a stochastic volume calculation when it is needed.

Optimal nuclear data management based on the NJOY code

The MCNP code goes with its associated standard libraries, filled with various ACE (A Compact ENDF) nuclear data files. The MURE user can also generate himself the data files that he needs by means of MURE/ENDF2ACE [2], a friendly C++ interface to the nuclear data processing code NJOY. NJOY produces the ACE file from ENDF data, and can compute thermal effects such as thermal scattering in moderators or Doppler effect.
MURE/ENDF2ACE allows thus to have as much different temperatures taken into account as needed, which is useful for precise temperature coefficients calculations for example. This feature makes easier benchmarks and nuclear data base comparisons too.
Within MURE are fully implemented a few tools dedicated to the easy management of the user personal nuclear data files.

Principles of the MCNP-based evolution calculations

Fuel evolution is based on a coupling between a static MCNP run and the resolution of nuclei evolution equations. Thus before any evolution, a full tree of nuclei is built according to nuclear data (decays and available nuclear cross-sections).
MURE allows building from each initial material composition a complete map of the nuclei (the nuclei tree) to be followed in an evolution calculation (at the beginning, only input nuclei have a non zero proportion). In order to reduce the total number of nuclei that evolve, the tree is simplified by means of a few physical criteria, such as a minimal half-life for decays and an integral cross-section threshold for nuclear reactions.

The main steps, summarized on Figure 1, can be decomposed as follows:

- The Nuclei tree is built once and for all at the first MCNP run ; Initial compositions of all materials are entered by the user and these will evolve automatically.
- All the necessary tallies for calculating mean neutron fluxes and cross-sections are automatically built in independent evolving cells (to take into account spatial flux heterogeneity).
- MCNP input file with the composition at a given time t_i is built and a MCNP run is performed.
- Bateman equations are solved by a standard 4^{th} order Runge-Kutta method using fluxes and cross-sections of MCNP run over a given Δt_i time interval.
- A new MCNP file with the composition at $t_{i+1} = t_i + \Delta t_i$ is performed, and so on.

There are three levels of time steps in the evolution calculation scheme :

- The first level is the one of MCNP runs. The length of these MCNP steps Δt_i are user-defined and are generally not regular, in order to optimize CPU time use.
- The second level is the discretization within a MCNP step. Each Δt_i is divided in N_{RK} equally spaced Runge-Kutta δt steps. At each $t_k = k.\ \delta t$, Bateman equations are built with mean reaction rates ; these reaction rates depend on time (because of the flux shape evolution) and an extrapolation is performed based on previous values. Special control methods (such as compensation of reactivity loss due to burn-up or flux renormalization to keep constant the total thermal power) can be called at these times. These methods apply to the concerned evolving cells.
- The last discretization step is performed automatically by the adaptative step size Runge-Kutta method. During each such elementary step, cross-sections as well as fluxes are kept constant.

2. First applications and results obtained by MURE on precise reactor simulations

Benchmarking of Fast Breeder Reactors

As an example of a complex geometry, the European Fast Reactor, which we are currently studying, is presented on figure 3. During the evolution, the control rods are automatically moved to keep reactivity constant through their worth curve.

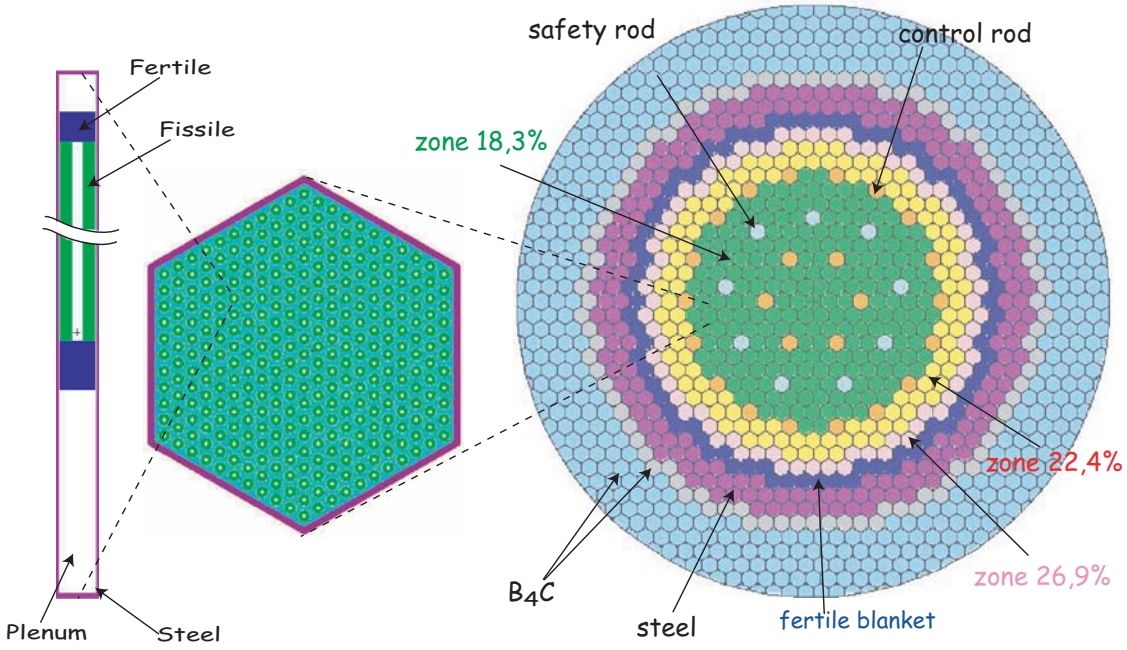

Figure 3: EFR geometry: from core scale to fuel pin scale.

A benchmark of the Russian Sodium Fast Reactor, BN 800 has been done by MINATOM, CEA and EDF[4] using deterministic codes. Results of a comparison with MURE/MCNP calculations are given in Table 1.

Table 1: BN 800 Benchmark

	Minatom	CEA(eranos)	MURE
k_{eff}	1.00270	0.99011	0.99640
Void coefficient (core)	1517	1580	1695
Void coefficient (reactor)	350	658	1034

These results show a pretty good agreement between Eranos and MURE/MCNP.

Study of U-233 production in Th/Pu loaded PWR[5]

The first realistic test of MURE evolution capabilities has been a simulation of the irradiation of Th/Pu fuel in a classical LWR assembly. The motivation for this study is the investigation of a transition scenario over the next century from a park of light water reactors to a park of Thorium Molten Salt Reactors, which require an initial inventory of ^{233}U produced upstream. In this example, the PWR geometry was defined in MURE, Different initial concentrations of Plutonium were investigated to obtain a negative void coefficient at any time. The highest concentration of Pu retained is 9.5%w, with a corresponding boron concentration in water of 550 ppm. The initial Pu isotopic vector corresponds to typical MOX Pu, obtained after a 50 GWd/t burn-up of UOX fuel (fissile concentrations: ^{239}Pu 56.3%, ^{241}Pu 7.5%).
MURE evolution was then performed on the assembly, using the boron control method that adjusts the boron concentrations in order to maintain the k_∞ of the assembly at 1.017 (corresponding to a core k_{eff} value of 1.0). The goal was to find the amount of ^{233}U production in a LWR assembly, configured in this way, at the end of the burn-up, and in addition the isotopic vector of the remaining degraded Plutonium. The results of the MURE evolution are shown in figure 4.

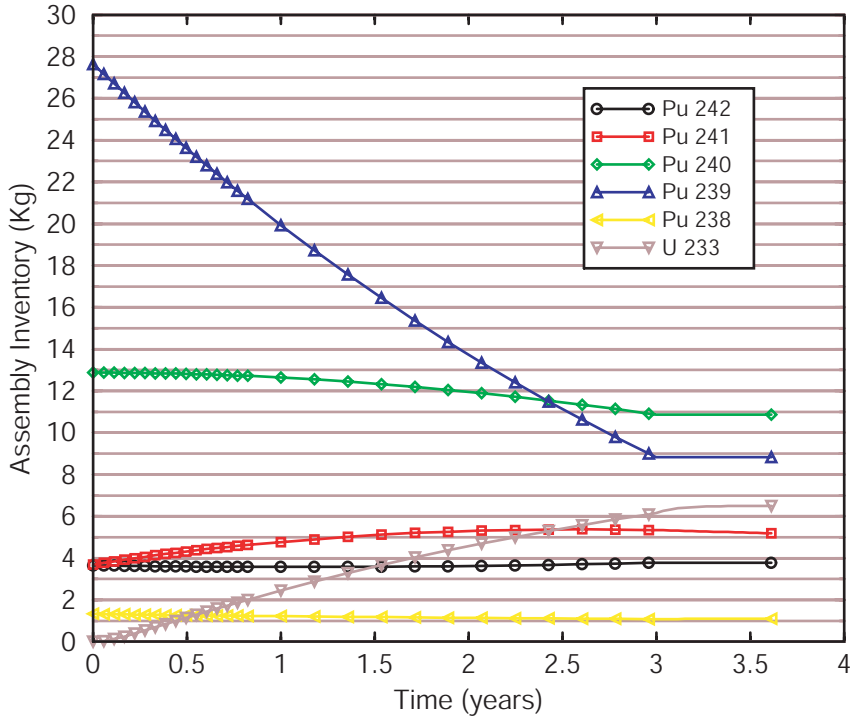

Figure 4: Evolution of main nuclides after 3 years (52 GWd/t burn-up) in core of a Th/Pu assembly and 6 months of cooling. Each dot corresponds to a MCNP run for the in-core period.

It can be seen (Fig. 5) that at the end of burn up (approximately 3 years) all of the boron concentration in the water moderator has essentially dropped to zero. At this point, the fuel must be removed. From the evolution calculations it can be seen that a final inventory of around 6 kg of ^{233}U is produced in the assembly, which rises to almost 6.5 kg after 6 months of further "cold" evolution (zero neutron flux) due to decay of the ^{233}Pa.

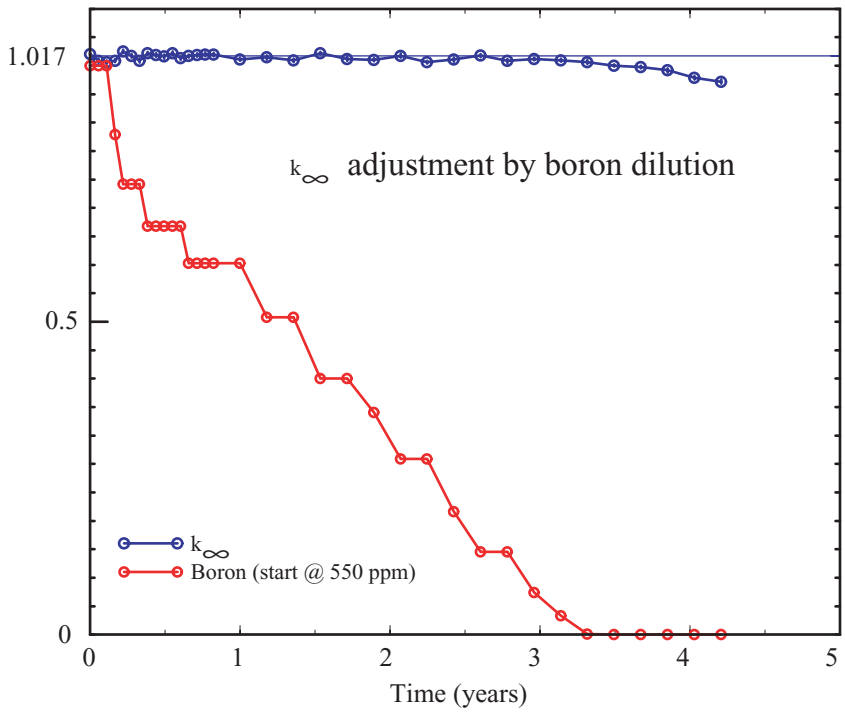

Figure 5: Evolution of K_∞ shown with normalized boron concentration.

Temperature calculation by coupling to a thermal-hydraulics code

The complete study of any Generation IV reactor must include neutronics, hydraulics and thermal effects. Coupled simulations of neutronics and hydraulics have already been performed for an experimental test case; the Molten Salt Reactor Experiment (MSRE), operated at the ORNL in the sixties [6]. Two reference 3D codes are used; the Monte Carlo code MCNP (developed by Los Alamos) for neutronics, by means of MURE, and the Finite Volume code TRIO-U (developed by CEA [7]) for hydraulics. At zero power, our simulations precisely reproduce experimental results such as the effects of fuel-pump start-up. This success leads us to study the same system under full power conditions, by adding the thermal treatment part of TRIO-U. Neutron data are computed for each core region at its right temperature thanks to the nuclear data processing code NJOY, by means of MURE again. An iterative scheme of alternate MCNP and TRIO-U runs allows us to determine equilibrium power and temperature spatial distributions. MCNP uses a precise geometric description of the reactor to compute local volumic powers in fuel salt channels and graphite clads, with a user-defined spatial resolution. TRIO-U computes then the heat exchanges in each core region by means of a channel model with corresponding volumic powers and fuel salt velocities. This method has been validated on some MSRE experimental results. Figure 6 gives a temperature profile obtained within a 1GWe power Thorium Molten Salt Reactor (TMSR), simulated with MURE.

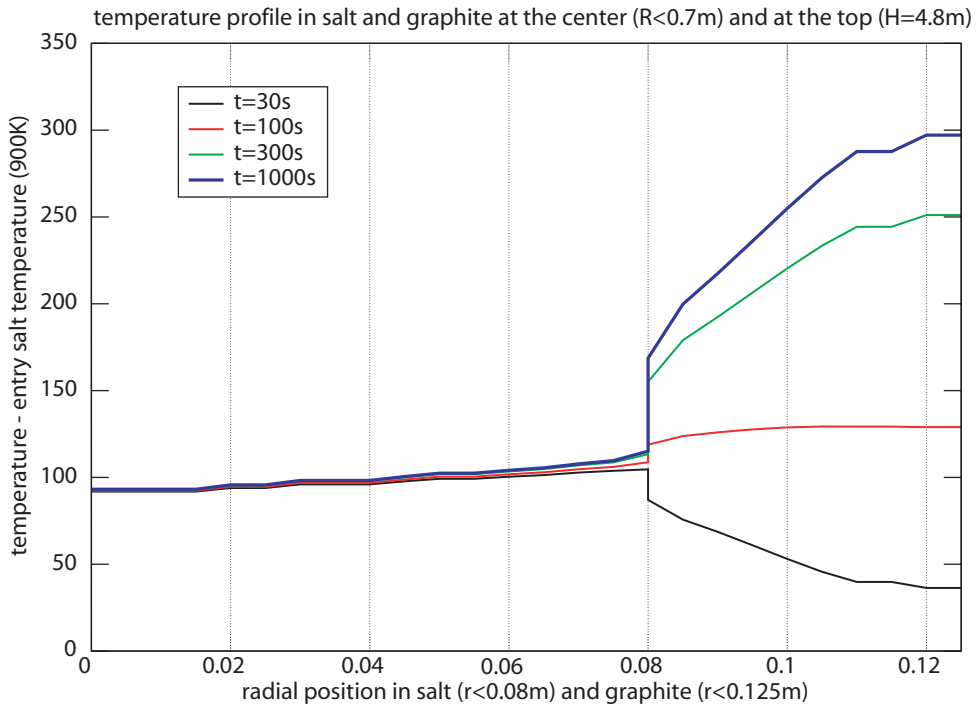

Figure 6: Result from a MURE-based coupled calculation using TRIO-U for thermal-hydraulics in a power Thorium Molten Salt Reactor turbulent channel. The temperature profile is given for a few times from initial conditions, until it reaches the equilibrium one (highest values).

Conclusions and perspectives

MURE consists of a powerful ensemble of utilities in C++, which are easy to use and available on http://lpsc.in2p3.fr/gpr/MURE/html/MURE/MURE.html. It has been constructed for numerical simulation of reactor evolution, experimental setup modelization, and especially for Evaluation of Future Reactor Designs. Up to now, MURE contains about 18000 lines of code in 20 different classes, and is in continuous development with a constantly increasing number of users. There is still some work to do, concerning among others evolution control features, graphical interface and coupling with a thermohydraulic code like TRIO-U for safety transient studies. Next studies will include new reactor simulations (for thorium cycle, in both existing reactors and generation IV projects, and for transmutation and incineration, in FNR and ADS) as well as scenario studies involving the studied systems.

Acknowledgements

The authors wish to acknowledge the valuable contributions of Fabien Perdu and Luc Perrot to the very first MURE development efforts.

References

[1] *MCNP - A General Monte Carlo N Particle Transport Code*, LANL, report 12625-M (1997)
|2] L. Perrot, *ENDF2ACE, User Manual,* IPN Orsay (2004)
[3] *NJOY97, Code System for Producing Pointwise and Multigroup Neutron and Photon Cross Sections from ENDF/B-VI Data*, RSICC CODE PACKAGE PSR-368 (1998)
[4] *Benchmark MINATOM/CEA/EDF*, internal report, HI-27/04/026/A
[5] F. Michel-Sendis et al., *Plutonium an Uranium 233 production in Thorium fueled LWR,* Global 2005, Tsukuba, Japan, October 9/13.
[6] F. Perdu, *Contributions aux etudes de surete pour des filieres innovantes de reacteurs nucleaires,* PhD thesis, UJF (2003)
[7] P. Ledac and L. Delapierre, *User Manual, TRIO-U V1.2*, SMTH/LDTA/2000-14 (2000)